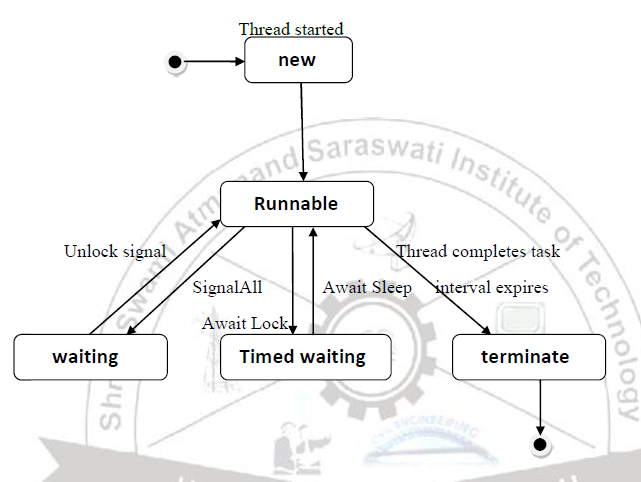
|  |  |
| --- | --- |
| **Thread** | **Process** |
| Threads (Light weight Processes) share the address space of the process that created it | processes have their own address. |
| Threads have direct access to the data segment of its process | processes have their own copy of the data segment of the parent process |
| Threads can directly communicate with other threads of its process | processes must use inter process communication to communicate with sibling processes |
| Threads have almost no overhead | processes have considerable overhead. |
| New threads are easily created | new processes require duplication of the parent process. |
| Threads can exercise control over threads of the same process | processes can only exercise control over child processes |
| Changes to the main thread (cancellation, priority change, etc.) may affect the behavior of the other threads of the process | changes to the parent process do not affect child processes |
|  |  |
|  |  |

Thread LifeCycle:



Thread has Five different states:

 **New:** A new thread begins its life cycle in the new state. It remains in this state until the program starts the thread. It is also referred to as a born thread.

 **Runnable:** After a newly born thread is started, the thread becomes runnable. A thread in this state is considered to be executing its task.

 **Waiting:** Sometimes a thread transitions to the waiting state while the thread waits for another thread to perform a task.A thread transitions back to the runnable state only when another thread signals the waiting thread to continue executing

 **Timed waiting:** A runnable thread can enter the timed waiting state for a specified interval of time. A thread in this state transitions back to the runnable state when that time interval expires or when the event it is waiting for occurs.

 **Terminated:** A runnable thread enters the terminated state when it completes its task or otherwise terminates

Write the different way to create thread using java.

By Implementing Runnable interface

To create thread using Runnable interface, a class first need to instantiate an object of type Thread from within that class. Thread defines several constructors. The one that we will use is shown here:

**Thread(Runnable threadOb, String threadName);**

Here threadOb is an instance of a class that implements the Runnable interface and the name of the new thread is specified by threadName.

After the new thread is created, we need to start the execution of thread.It is done using its **start( )** method, which is declared within Thread.

**void start( );**

we can define the code that constitutes the new thread inside run() method. run() method can call other methods, use other classes, and declare variables, just like the main thread can.

**public void run( )**

By extending the Thread class

To create a new thread first need to extend **Thread** Super class and create an instance of that class. The newly created(extended) class must override the **run()** method, which is the entry point for the new thread. It must also call **start( )** to begin execution of the new thread. Same as Runnable Interface